Remarks/Arguments

A. Pending Claims

Claims 14-16 are new. Claims 11-13 have been amended. Claims 1-16 are pending in the case.

B. Abstract

The Examiner objected to the abstract of the disclosure as containing more than 150 words. The abstract has been amended to include less than 150 words.

C. The Claims Are Not Anticipated By Brandt Under 35 U.S.C. §102(b)

The Examiner rejected claims 1-3, 5-7, and 10-12 under 35 U.S.C. §102(b) as being anticipated by U.S. Patent No. 5,892,905 to Brandt et al. (hereinafter "Brandt"). Applicant respectfully disagrees with these rejections.

The standard for "anticipation" is one of fairly strict identity. To anticipate a claim of a patent, a single prior source must contain all the claimed essential elements. *Hybritech, Inc. v. Monoclonal Antibodies, Inc.*, 802 F.2d 1367, 231 U.S.P.Q. 81, 91 (Fed.Cir. 1986); *In re Donahue*, 766 F.2d 531, 226 U.S.P.Q. 619, 621 (Fed.Cir. 1985).

Claims 1, 5, and 10 describe combinations of features including "accessing a first database in response to writing the first database identifier value into the third memory." Brandt does not appear to teach or suggest at least this feature of claims 1, 5, and 10, in combination with the other features of the claims.

The portion of Brandt cited for the above-quoted feature of claims 1, 5, and 10 states:

> In this case, since the car rental agent has submitted a request to start a work item, the value of the submit variable is equal to 3. CGI 420 will retrieve the specified template (in this case exmp5ewi.htm), parse the template for HTML variables, and pass these variables along with the data stream, environment data and control information to FMIG 430. While control passes to FMIG 430 at this point, CGI 420 remains connected to FMIG 430 and waits for data to be returned from FMIG 430. FMIG 430 authenticates the car rental agent using the variables transmitted by CGI 420. Since the car rental agent is already logged onto FlowMark 450, FMIG 430 need not log the car rental agent on again. FMIG 430 parses the data stream; sees the request to start a work item, (wf-api-item) and issues a FlowMark API 436 to start the processing the work item. Since the work item has been transmitted in a WWW context with HTML variables, FMIG 430 knows the activity program will use WWW APIS 434 to have a conversation with the web client. FMIG 430 will generate a handle for this web client and will enter the handle, process instance name, activity name, userID, etc. into the internal cache. FMIG 430 also saves the HTML variable information so that it can be restored later when the conversation has taken place. FMIG 430 will then wait for WWW APIs 434 to connect.
> (Brandt, column 31, lines 26-49).

Brandt appears to teach a "FlowMark/Internet Gateway (FMIG)" that issues an application program interface to start processing of a work item. An activity program uses WWW application program interfaces to have a conversation with a web client. Brandt does not appear to teach or suggest the above-quoted features of claims 1, 5, and 10. The Office Action states: "it is inherent that the HTML has to be written and the internal cache is interpreted as the third memory." Applicant respectfully disagrees with the Office Action's position. Moreover, to rely on the theory of inherency, the examiner must "provide a basis in fact and/or technical reasoning to reasonably support the determination that the allegedly inherent characteristic <u>necessarily</u> flows from the teachings of the applied prior art." *Ex parte Levy*, 17 U.S.P.Q.2d 1461, 1464 (Bd. Pat. App. & Inter. 1990) (emphasis in original.) Inherency may not be established by probabilities or possibilities; the mere fact that a certain thing <u>may</u> result from a given set of circumstances is not sufficient. *In re Robertson*, 169 F.3d 743, 745 (Fed. Cir. 1999). Applicant submits that Brandt does not <u>necessarily</u> include writing a database identifier value into a third memory. In any case, Brandt does not appear to teach or suggest accessing a first database in response to writing a first database identifier value into a third memory, nor do these features appear to be inherent to the teachings of Brandt.

Claims 1, 5, and 10 also describe "wherein the one or more key values and the one or more database identifier values stored in the second memory are entered by a user of the FSO computer system during a configuration of the FSO computer system." Brandt does not appear to teach or suggest at least this feature of claims 1, 5, and 10, in combination with the other features of the claims.

The Office Action cites a lengthy portion of Brandt (column 28, lines 57 through column 30, line 44) for the above-quoted feature of claims 1, 5, and 10. The cited portion appears to disclose a process for renting a car by allowing the customer to access workflow management application software (namely, "FlowMark"). The cited portion does not appear to disclose entry of key values and one or more database identifier values stored in a second memory. Moreover, the cited portion of Brandt appears to relate actions taking place during use of a system to rent a car (e.g., to confirm a reservation) to a customer, not during configuration of an FSO computer system. For example, Brandt states:

> During the processing of the reservation confirmation template, activity program 432 has issued a Receive API to WWW APIs 434 which acts as a confirmation message for activity program 432 to ensure that the customer has received and viewed the confirmation number. The web client then clicks on the "submit" button when they have received and recorded their confirmation number. This data is transmitted, as before to web server application 222 along with the web userID and password which have been retained by web browser 212. Web server application 222 uses this information to once again authenticate the web client to CGI 420.
> (Brandt, column 28, line 57 through column 29, line 1)

> FMIG 430 then retrieves the stored variable values from the internal data cache and transmits the data to CGI 420. CGI 420 receives the data and variables and detaches from FMIG 430. CGI 420 then contacts web server application 222 and transmits the variables and data and directs the web server application to render the specified HTML to web browser 212.
>
> At this point, interaction with the user regarding the car rental request has been completed.
> (Brandt, column 29, lines 28-36)

For at least the reasons provided above, Applicant submits that claims 1, 5, and 10 and the claims depending from claims 1, 5, and 10 are patentable over the cited art. Applicant respectfully requests the removal of the 35 U.S.C. §102(b) rejections of these claims.

In addition, Applicant submits that claims dependent on claim 1 are independently patentable. For example, claims 2, 6, and 11 describe "wherein the accessing the first database in response to writing the first database identifier value into the third memory comprises: comparing the first database identifier value in the third memory to the active database identifier value in the fourth memory." Applicant respectfully submits that this feature is not taught or suggested by the cited art.

The Office Action takes the position that Brandt discloses the above-quoted feature of claims 2, 6, and 11. Applicant respectfully disagrees with this position. The portion of Brandt cited in the Office Action states:

> Activity program 432 will use FlowMark APIs to retrieve any input data from the FlowMark input data container. The activity program will open a conversation with the web client by issuing, an Open API to WWW APIs 434. The Open API generates an Open request message for FMIG 430 and includes information such as the process instance and activity name and FlowMark userID. FMIG 430 takes this information, matches up the web client by finding the appropriate process instance, activity name, and userID in the internal data cache. As before, FMIG 430 also generates a conversation identifier and stores the

identifier in its internal cache. FMIG 430 transmits the conversation identifier to WWW APIs 434 in a response message and the WWW APIs 434 detach from FMIG 430 and return the conversation identifier to activity program 432. From this point on, the conversation identifier is included in all WWW API 434 submissions dealing with this web client and this process instance. The conversation identifier remains valid until the conversation is closed. Activity program 432 issues a Send API to WWW APIs 434 to generate an HTML screen to the web client. Activity program 432 specifies the data type as HTML template so that CGI 420 knows to parse the template and replace the variables with the data sent by activity program 432. Activity programs 432 specify the location of the template and the text to be substituted for each variable.

The Send API generates a send request to FMIG 430 and sends the data. FMIG 430 matches the included conversation identifier with the appropriate web client and transmits the data from the activity program 432, the data type and the handle to the still-attached CGI 420. At this point, CG1 420 detaches from FMIG 430 and starts to process the received data. FMIG 430 responds to the WWW API 434 and WWW API 434 also detaches from FMIG 430. WWW APIs 434 send a return code to activity program 432. CGI 420 will use the data transmitted from FMIG 430 to create a screen containing information about cars that are available to fill the customer's request. In addition, since CGI 420 is in communication with web server application 222, CGI 420 will insert a hidden variable, the "wf-fmig-handle", and the wf-fmig key from the first HTML screen into the HTML code for the new screen. CGI 420 then parses the template and processes the "wf-" variables, filling information into the appropriate locations. The HTML code in the template file would look similar to the HTML code shown in FIG. 21. After CGI 420 finishes processing, the HTML code in the file would look like the HTML code in FIG. 22. This is the HTML code that web server application 222 will use to render the car rental agent's screen on client workstation 210. In the meantime, activity program 432 has issued a Receive WWW API 434 to receive the data from the web client. The car rental agent selects which car to reserve in order to fill this car rental request and clicks on the "submit process" button. The communication process between web browser 212, web server application 222, CGI 420, FMIG 430 and FlowMark 450 takes place as explained above. Eventually, the selected car will be updated in FlowMark database 438 and the conversation will be terminated. Since the car rental reservation process instance is now complete, FlowMark 450 removes the process instance from FlowMark database 438.
(Brandt, column 31, line 58 through column 32, line 51)

Brandt appears to disclose a FlowMark/Internet Gateway (FMIG) generating a "conversation identifier." The conversation identifier is included in all WWW application program interface submissions dealing with a particular web client and a particular process instance. Brandt does not appear to teach or suggest wherein accessing a first database in response to writing a first database identifier value into the third memory comprises comparing the first database identifier value in the third memory to an active database identifier value in a fourth memory.

Claims 2, 6, and 11 also describe "setting the active database to the first database in response to the first database identifier value in the third memory not matching the active database identifier value in the fourth memory." Applicant respectfully submits that this feature is not taught or suggested by the cited art.

The Office Action takes the position that Brandt discloses the above-quoted feature of claims 2, 6, and 11. Applicant respectfully disagrees with this position. The portion of Brandt cited in the Office Action states:

> Activity program 432 checks the return code and verifies that the disconnect API was successfully recorded. Activity program 432 saves the conversation identifier and status information in a local database for restoring the disconnected process at a later time. Since this activity has progressed as far as possible without actually completing, control is returned to the PEC. Since the activity did not complete, the PEC will update FlowMark database 438 to show that the activity is ready to be started again. As far as FlowMark is concerned, this is a "manual start" activity. Since this activity is ready and manual start, FMIG 430 can issue a FlowMark API 436 at a later time to re-start the activity when the data is available from the car rental agent. The car rental agent selects that they wish to move a car from an alternate location to fill the car rental request and clicks on "submit." The process of data transfer and web client authentication proceed as previously described. This time, when FMIG 430 checks the conversation identifier, it will notice that the status of the activity is "disconnected." FMIG 430 issues FlowMark API 436 to re-start the activity program 432. Activity program 432 is thus re-started FlowMark 450 creates the process instance and updates database 438 as before. The PEC starts the activity program 432 and the necessary data is transferred from FMIG 430. Activity program 432 can now run to completion. It is possible to determine which rental

> car should be moved from which location and to transmit this data from the car rental agent to FlowMark 450 by a similar sequence and series of communications.
> (Brandt, column 33, lines 2-31)

Brandt appears to disclose an activity program used in moving a rental car from one location to another. A "FlowMark/Internet Gateway (FMIG)" issues an application program interface to re-start the activity program, whose activity has previously been "disconnected." Brandt does not appear to teach or suggest setting an active database to the first database in response to a first database identifier value in a third memory not matching an active database identifier value in a fourth memory.

Claims 3, 7, and 12 describe "wherein setting the active database to the first database comprises setting the active database identifier value stored in the fourth memory to the first database identifier value from the third memory." Applicant respectfully submits that this feature is not taught or suggested by the cited art.

The Office Action takes the position that Brandt discloses the above-quoted feature of claims 3, 7, and 12. Applicant respectfully disagrees with this position. The portion of Brandt cited in the Office Action states:

> Next, once a valid customer identity has been established, another activity program 432 may be initiated which will determine if the request can be filled according to the customer's request. This would be encompassed in process step 2020 of FIG. 23. For example, is a car available on the requested date, in the requested city, in the requested size, etc.? [*sic*] Alternatively, the FlowMark process model may specify that the car rental request should be routed to a human agent for further processing. In that case, the car rental request would show up on a FlowMark task list for the agent. Alternatively, the entire process may be completely automated. In either case, the car rental agent or activity program 432 processes the web client's car rental request and if the desired car is available, FlowMark database 438 will be updated to indicate that the car has been "reserved." These activities would be accomplished in process step 2030 of FIG.

23. If, however, the desired car is not available, a new activity program 432 may be initiated to look for a car in an alternate location (process step 2040 of FIG. 23) and create a request from a human car rental agent to transfer the desired automobile from another location to the desired location (process step 2050 of FIG. 23). Some of these activity programs 432 are described in greater detail below.

In this example, once the car rental agent has approved the request and transferred the vehicle to the desired location, the car rental agent will provide an input to FlowMark application 342 and update the activity program 432. Once the rental car has been reserved in the system, a different activity program 432 may be initiated to generate a confirmation number for the customer and process the confirmation transaction with the customer. Each activity program 432 is designed to be an independent process which executes to conclusion and then quits.

At this point, activity program 432 uses FlowMark APIs 436 to retrieve any input data from the FlowMark input data container. The FlowMark data container is a FlowMark function that is defined when a FlowMark process model 440 is built. The FlowMark data container is accessible via FlowMark APIs 436 and is used as a storage location to store and pass status and information from one activity program 432 to the next activity program 432 in process model 440. Then, activity program 432 opens a conversation with the web client by issuing an Open to WWW API 434 via connection 428. Activity program 432 also includes information such as the process instance being executed, activity name, and FlowMark userID. This information is transmitted to FMIG 430 by WWW API 434. FMIG 430 matches the Open API with the appropriate web client by locating the requested process instance name in the internal data cache. FMIG 430 then generates a "conversation identifier" for this transaction. FMIG 430 saves the conversation identifier in its internal cache and transmits the conversation identifier to WWW API 434 as part of a response message. WWW API 434 detaches from FMIG 430 and returns the conversation identifier to activity program 432. From this point on, the conversation identifier is included on all WWW API 434 submissions between this web client and all activity programs 432 necessary to process the web client's request. The conversation identifier remains valid until the conversation is terminated by a Close API issued by activity program 432.
(Brandt, column 27, lines 5-64)

Brandt appears to disclose activity programs used to perform steps relating to rental of a car (e.g., determining whether a request can be filled, reserving a car, confirmation a reservation). A "conversation identifier" is included in WWW application program interface submissions between the activity programs and a web client. Brandt does not appear to teach or suggest setting an active database identifier value stored in a fourth memory to a first database identifier value from a third memory.

D. Additional Remarks

Based on the above, Applicant submits that all claims are in condition for allowance. Favorable reconsideration is respectfully requested.

If any extension of time is required, Applicant hereby requests the appropriate extension of time. If any fees are omitted or if any additional fees are required or have been overpaid, please appropriately charge or credit those fees to Meyertons, Hood, Kivlin, Kowert & Goetzel, P.C. Deposit Account Number 50-1505/5053-31201/EBM.

Respectfully submitted,

[illegible]

Eric B. Meyertons
Reg. No. 34,876

Attorney for Applicant

MEYERTONS, HOOD, KIVLIN, KOWERT & GOETZEL, P.C.
P.O. Box 398
Austin, TX 78767-0398
(512) 853-8800 (voice)
(512) 853-8801 (facsimile)

Date: 12-21-04